Comparison of Image Processing Resource Allocation for Multi-Target Tracking of Dismounted Targets

Jon P. Champion, Second Lieutenant, USAF
Distributed Information Systems Branch
Air Force Research Laboratory
525 Brooks Rd
Rome, NY 13441, USA
(315) 330-2709
jon.champion@rl.af.mil

Juan R. Vasquez, Lieutenant Colonel, USAF
Department of Electrical and Computer Engineering
Air Force Institute of Technology
2950 Hobson Way
WPAFB, OH 45433-7765, USA
(937) 255-3636 x7231
juan.vasquez@afit.edu

Abstract - *Dismounted targets can be tracked in urban environments with video sensors. Real-time systems are unable to process all of the imagery, demanding some method for prioritization of the processing resources. Furthermore, various segmentation algorithms exist within image processing, each algorithm possesses unique capabilities, and each algorithm has an associated computational cost. Additional complexity arises in the prioritization problem when targets become occluded (i.e., a building) and when the targets are intermixed with other dismounted entities. This added complexity leads to the question "which portions of the scene warrant both low cost and high cost processing?" The approach presented in this paper is to apply multi-target tracking techniques in conjunction with an integer programming optimization routine to determine optimal allocation of the video processing resources. This architecture results in feedback from the tracking routine to the image processing function which in turn enhances the ability of the tracker.*

Keywords: Tracking, resource allocation, image processing, urban dismount, integer programming.

1 Introduction

The operational impetus for the research presented here is the need to track dismounted targets in urban environments. Recent efforts by the Automatic Target Recognition (ATR) Division of the Air Force Research Laboratory Sensor's Directorate have led to the focus on video based sensing in order to accomplish urban dismount tracking. Two issues that naturally arise in these types of scenarios are multiple targets and regions of measurement occlusion (i.e., targets behind buildings). This paper addresses these specific issues by combining multi-target tracking with an image processing resource allocation algorithm. Details of the methods used are presented along with preliminary results demonstrated via computer simulation. To accomplish effective multiple-target tracking, computational complexity is affected by the complexity of the tracker. More complex systems may be more robust, but demand more resources from the computer. When using video-based tracking, an additional computational complexity arises from the measurement generation process commonly known as image processing. Real-time systems may be unable to process all of the imagery provided by the sensor. Furthermore, one has to decide which segmentation method to use on the image as each method possesses unique capabilities and each method has an associated computational cost. The focus of this research is the development of an optimization methodology to determine the allocation of the video processing resources. A key aspect of the architecture presented is the information exchange between the tracking function and the resource allocation function. Each function enhances the capability of the other, and a feedback loop can be formed for this application.

It is important to note that the tracking function consists of both kinematic and attribute tracking. Given the sensor mode, namely imagery generated by video, and the target type, namely a dismounted target, attributes will play a major role in the ability to maintain kinematic track and to provide target ID. Attribute measurement data may be provided by image segmentation methods such as color mapping (hue and

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
JUL 2006		**00-00-2006 to 00-00-2006**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Comparison of Image Processing Resource Allocation for Multi-Target Tracking of Dismounted Targets	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Air Force Research Laboratory,Distributed Information Systems Branch,525 Brooks Rd,Rome,NY,13441**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
9th International Conference on Information Fusion, 10-13 July 2006, Florence, Italy. Sponsored by the International Society of Information Fusion (ISIF), Aerospace & Electronic Systems Society (AES), IEEE, ONR, ONR Global, Selex - Sistemi Integrati, Finmeccanica, BAE Systems, TNO, AFOSR's European Office of Aerospace Research and Development, and the NATO Undersea Research Centre.

14. ABSTRACT
see report

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**8**	

saturation), texture mapping, and change detection. These three methods will be exploited in this paper, but the overall architecture presented is easily extendable to other methods. The attributes provided by color and texture mapping are obvious, and the specific method used in the simulation will be discussed in Section 3. Change detection can provide both kinematic and attribute data. The mass of a detection is considered an attribute, and the center of this mass provides kinematic position data. Further detail of the tracking algorithm will be presented in Section 4.

With this basic understanding of the measurement information used by the tracking algorithm, it is easy to recognize the interaction of the two primary functions. Specifically, the kinematic data is used by the resource allocation function to determine regions that should receive a higher allocation of the image processing resources. In addition, the ability of the resource allocation routine to provide the optimal set of attribute measurements to the tracking algorithm will ensure enhanced tracking.

The next sections provide details of the resource allocation method followed by a specific implementation of a relatively simple tracking algorithm and the version of the image segmentation methods used in the simulation study.

2 Optimal Resource Allocation

The fundamental questions being answered by the resource allocation algorithm are "which regions of the image should have priority for image processing?" and "which of the various segmentation algorithms should be applied in these regions?" A straightforward method is used by dividing the image into sub-regions at various levels of resolution. This results in a grid as illustrated in Figure 1 for a 21 sub-region case. In this example, the first 16 sub-regions represent the high resolution regions, 17-20 are medium resolution, and 21 has the lowest resolution. Various image segmentation algorithms may be employed in each sub-region. As such, the number of options available to the resource allocation process are the number of sub-regions multiplied by the number of segmentation methods. A key factor in choosing which options to execute is the benefit associated with each option. There are variations in the costs associated with processing high versus low resolution regions, and variations in the costs of each type of segmentation algorithm. Given these viewpoints, it is clear that a cost/benefit based optimization can be performed.

The value associated with each sub-region may depend on many factors. The primary factors in this study come from the kinematic states of the tracking routine, the relative value of the segmentation methods, and the target ID associated with a given track. The distance of a sub-region relative to a given track's position will be considered for the assignment of values to each sub-region. The full computation that determines the value of a given sub-region will be described in Section 4.4. The focus of this paper will be to establish the minimal valuation criteria to obtain a meaningful selection within the optimization routine. Noting that the values assigned to a sub-region will be based on the kinematic data gathered from previous measurements, it is clear that the information gained from a tracking algorithm will be used to drive the optimization routine. It is also evident that the competence of the algorithm that derives the values for sub-regions is also of critical importance to the system.

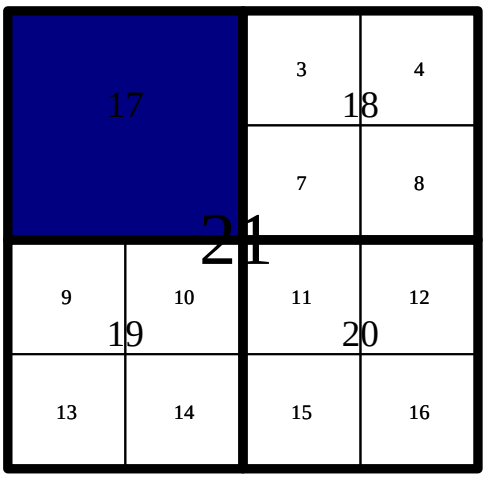

Figure 1: Sub-region Grid

Given the nature of the binary decisions being made an obvious choice is the use of binary integer programming to search this solution space. Linear integer programs can be solved by a number of conventional solution techniques so the formulation of the linear program is the critical issue [2]. As with the gridding and valuation considerations, the integer program must ensure that it enhances the capability of the system while reducing the overall computational load.

The primary constraint on the system is a limitation on how much time it should take to complete the entire process. With knowledge of the relative time requirements to complete the available segmentation tasks, the integer program can be devised to solve the problem. Other system limitations can be implemented, which may vary depending on the segmentation algorithms available. Constraints are described as arithmetic equations that can define lines or planes in a space. Each of the sub-regions in conjunction with the set of segmentation algorithms are considered a binary decision variable. Selecting a decision variable will add a value to the solution while incurring a cost. The optimal solution will be obtained when the maximum value has been obtained while consuming all of the available time to perform the chosen segmentation algorithms.

We formally define the binary integer program as follows:

$$\begin{array}{ccc} \text{maximize} & f_{i,j}d_{i,j} & \\ \text{such that} & Ad_{i,j} \leq b & \\ \text{and} & A_{eq}d_{i,j} = b_{eq} & \end{array} \quad (1)$$

where

$$\begin{aligned} d_{i,j} &= \text{binary decision variable for sub-region } i \\ &\quad \text{and segmentation method } j \\ f_{i,j} &= \text{value function at } i,j \\ A, A_{eq} &= \text{constraint matrices for limits on } d_{i,j} \\ b, b_{eq} &= \text{limit vector for the constraints} \end{aligned}$$

Note that $d_{i,j}$ is a vector of length equal to the number of image sub-regions across all resolution levels (n_{sub}) multiplied by the number of segmentation methods (n_{seg}) considered. A detailed description of the computation of the value function, $f_{i,j}$, is deferred until Section 4.4 due to its dependence on information presented later in the paper. The inequality constraints provide upper bounds on the number of decision variables with a value of 1 and thus limit the number of times that a sub-region and segmentation method can be selected. The structure of the A-matrix is such that the first row in conjunction with the first element of the limit vector, b, will provide a limit on the number of times segmentation method #1 can be performed. A similar row is added for each segmentation method. Next, a row is added for each sub-region in conjunction with another element of the vector b to provide a limit on the number of times each sub-region can be operated on at each sample time. The dimensions of the A-matrix will be $(n_{sub} + n_{seg})\,\mathrm{x}\,(n_{sub} * n_{seg})$.

The second set of constraints is used to ensure that the optimization ensures all of the processing time available at each epoch will be used. This is accomplished by setting b_{eq} equal to the maximum allowable cost which represents the total processing time available. The elements of A_{eq} are based on the time required to perform each of the possible segmentation tasks. The form of A_{eq} is given by:

$$\left[\; c_1 \;\; \overset{n_{sub}}{\cdots} \;\; c_1 \;\; c_2 \;\; \overset{n_{sub}}{\cdots} \;\; c_2 \;\; c_{n_{seg}} \;\; \overset{n_{sub}}{\cdots} \;\; c_{n_{seg}} \;\right]$$

where c_s = the cost to perform segmentation method s. The dimensions of the A_{eq}-matrix will be 1 x $(n_{sub} * n_{seg})$.

The binary integer program is itself a computationally intensive mechanism. To ensure that this method does not merely trade one numerically intensive mechanism for another, there are means that can be employed to reduce the overall size of the integer program. A form of gating in which sub-regions may be chosen based on their likelihood of providing information may be used. These sub-regions would be the only ones available for consideration within the integer program. This method would require increased confidence in the tracker and assumes that new targets are not entering the image space. This may be accomplished by applying wide area change detection at low resolution, then executing the integer program given the resources that remain.

3 Image Processing

Various segmentation methods can be applied to 2-D imagery for the purpose of extracting measurements that are processed by a tracking algorithm. Although this study focuses on three such methods, it is important to note the resource allocation function and the overall tracking architecture can readily accept any number of segmentation methods. The three methods mentioned previously are change detection, color mapping, and texture mapping.

Change detection is computationally inexpensive and is a pixel by pixel binary indication that a change has occurred in the image. The fundamental concept is to compare frames of imagery and detect variations from one frame to the next. Simply declaring a change based on two frames of data is susceptible to noise and stationary objects with second order movements. Multi-frame smoothing may be applied to filter out these movements, and focus on moving objects of greater interest to the tracker. Given that the change detection of a group of contiguous pixels defines a moving object, there are two main measurements that can be utilized. First, the mass of the object is defined by the number of pixels in the group and thus provides an attribute type measurement. Second, the center of mass of the pixels defines the kinematic location of the object. Both of these measurements will be provided to the tracking algorithm as described in the next section.

Color mapping uses a combination of hue and saturation levels for an object within a sub-region. A finite discrete number of bins, n_{bins}, is established to represent the possible variations of color. For this study, $n_{bins} = 30$, resulting in a set of 30 distinguishable hue levels, and 30 more bins are used to distinguish saturation levels. Each group of bins represents an attribute measurement. The computation of hue and saturation values are done pixel-by-pixel using the transformation [3]:

$$hue = \cos^{-1}\left\{\frac{1}{2}\frac{[(R-G)+(R-B)]}{\sqrt{(R-G)^2+(R-B)(G-B)}}\right\} \tag{2}$$

$$sat = 1 - \frac{3\min(R,G,B)}{(R+G+B)} \tag{3}$$

where R, G,and B represent the values of red, green, and blue present in each pixel. This will provide a value from 0 to 1 for each pixel for both hue and saturation. The value space is discretized into 30 evenly-spaced bins for this experiment. The same binning process is used for saturation data.

The intensity component of an image has the characteristic of being useful for describing an image's texture. Intensity is defined as:

$$I = \frac{1}{3}(R+G+B) \tag{4}$$

The spatial characteristics of texture can be calculated using the Fourier transform. This provides the ability to distinguish between periodic and non-periodic patterns and to quantify differences between periodic patterns. Spectral measurements are a result of the 2-dimensional Fast Fourier Transform (FFT) of the

image. This FFT produces a function of the image, $S(r,\theta)$, in polar coordinates. Two 1-dimensional summations of this function provide two additional attributes through:

$$S(r) = \sum_{\theta=0}^{\pi} S_\theta(r) \quad (5)$$

$$S(\theta) = \sum_{r=1}^{R_0} S_r(\theta) \quad (6)$$

where R_0 is the radius of a circle centered at the origin.

4 Target Tracking

As previously mentioned, the method employed to track targets is a combination of kinematic and attribute tracking. Kinematic tracking here assumes a 2-D space within which the targets may move. An occlusion is defined as a subset of the target space where measurements are not available, but where the target may move freely. States used to represent the kinematic and attribute data can be decoupled based on the following discussion. Attributes including color, texture, and mass do not effect the target dynamics model. In many cases, attributes lead to target ID, which may be correlated to target dynamics. However, our only targets of interest are urban dismounts, so the target dynamics are assumed identical. Similarly, target position and velocity will not directly effect target color, texture, or mass. This decoupling provides considerable simplification for the target tracker.

4.1 Kinematic Tracking

Consider the kinematic states of position and velocity $\mathbf{x}_{\text{kin}}(k) = [p_x \ v_x \ p_y \ v_y]^T$, the constant velocity model is given by according to the following constant velocity dynamics model [4]:

$$\begin{aligned} \mathbf{x}_{\text{kin}}(k) &= \Phi(k,k-1)\mathbf{x}_{\text{kin}}(k-1) + \quad (7)\\ &\quad +\mathbf{G}_d(k-1)\mathbf{w}_d(k-1) \\ &= \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{x}_{\text{kin}}(k-1) + \quad (8)\\ &\quad + \begin{bmatrix} \frac{T^2}{2} & 0 \\ T & 0 \\ 0 & \frac{T^2}{2} \\ 0 & T \end{bmatrix} \mathbf{w}_d(k-1) \end{aligned}$$

where T is the time between measurement points $(k-1)$ and k. The measurements consist of a two-element vector related to the state vector by the expression:

$$\begin{aligned} \mathbf{z}_{\text{kin}}(k) &= \mathbf{H}\mathbf{x}_{\text{kin}}(k) + \mathbf{v}(k) \quad (9)\\ &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \mathbf{x}_{\text{kin}}(k) + \mathbf{v}(k) \end{aligned}$$

where T is the time between measurement intervals $(k-1)$ and (k), and $\mathbf{w}(k)$ and $\mathbf{v}(k)$ are two independent zero-mean white noise processes such that:

$$\begin{aligned} E\{\mathbf{w}(t)\mathbf{w}(t+\tau)^T\} &= \mathbf{Q}\,\delta(\tau) \\ E\{\mathbf{v}(t_i)\mathbf{v}(t_j)^T\} &= \mathbf{R}\,\delta_{ij} \end{aligned}$$

Given this simple target model, linear Kalman filtering will be used for the kinematic tracker. The classical method of state propagation between measurement updates is conducted, while measurement updates may not occur at every sample time due to occluded targets. As such, during periods of prolonged occlusion, the target track errors will grow. The reappearance of the target on a boundary of the occlusion would provide a measurement update for the existing track. However, there is no guarantee that the target will reappear by the time the propagated track reaches a boundary of the occlusion. This leads the to need to manage track initiation and deletion. Candidate tracks are initiated via change detections at locations equal to the center of mass of the detection. A goal of the tracking routine is to maintain track of any high-valued targets. More than one track may exist since we cannot assume that the first detected change represents a high-valued target. This essentially represents multiple hypotheses in the context of many candidate tracks competing for the role as the high-valued target. Note again that the resource allocation concept is not dependent on the idea of a high-valued target, and it could still provide optimal processing decisions for multiple high-valued targets. Standard tracking concepts such as track gating and M/N initiation may help to filter out unrealistic new tracks. Track deletion techniques such as track scoring may be used to remove tracks of targets that are believed to have left the scene.

4.2 Attribute Tracking

The attribute state vector represents a concatenation of four types of attribute data as follows:

$$x_{\text{att}} = \begin{bmatrix} hue_{1-n_{bins}} \\ sat_{1-n_{bins}} \\ S(r) \\ S(\theta) \\ mass \end{bmatrix} \quad (10)$$

where $hue_{1-n_{bins}}$ and $sat_{1-n_{bins}}$ are vectors of dimension n_{bins} x 1 representing the 30 bins of hue and saturation data respectively. The remaining attributes are scalars as defined in Section 3. Inadequate knowledge of the statistical nature of the target state and the measurements leads to a desire to use a sub-optimal approach to perform attribute updates. Measurement updates are executed based on the simplistic and understandably ad hoc method of a mixture weighting given by [5]:

$$x_{\text{att}}(k) = \alpha x_{\text{att}}(k-1) + (1-\alpha) z_{\text{att}}(k) \quad (11)$$

where the weight α is empirically chosen. The attribute states are initialized with the first measurement of each type, and those attributes lacking any measurements take on a null value to prevent an erroneous target ID.

4.3 Target ID

Target ID is mapped to three discrete values: $Desired, Decoy, Unknown$. The features of the desired target, denoted here as "true", must be set a priori or determined by prolonged observation. Bayesian inference is used to compute the probability of ID, $P_{ID} = p(x_i|V_k)$, for each track as follows [6]:

$$\begin{aligned} p(c_i|V_k) &= \frac{p\left(V_k|c_i\right)p(c_i)}{\sum_j p\left(V_k|c_j\right)p(c_j)} \\ &= \frac{p\left(v_k|c_i\right)p(c_i|V_{k-1})}{\sum_j p\left(v_k|c_j\right)p(c_j|V_{k-1})} \end{aligned} \tag{12}$$

where the discrete transitional density, $p\left(V_k|x_i\right)$, is based on attribute data using a simple binary voting method that represents the set intersection of the various features. Note that Eq. (12) assumes independence of the measurements in the sense that:

$$p\left(V_k|c\right) = p\left(v_k, v_{k-1}, ..., v_1|c\right) = \prod_{i=1}^{k} p(v_i|c) \tag{13}$$

The declaration of target ID is made using maximum a posteriori (MAP) inference at a given sample time via [6]:

$$ID = \arg\left\{\max_i p(c_i|V_k)\right\} \tag{14}$$

This hard decision is acceptable for many applications. It answers the question of "What class of target is it?" Given the four types of attributes defined previously, a target can receive 0 to 4 votes to determine if the target matches the features of the desired target. The transitional density, $p\left(v_k|x_i\right)$, will take on discrete values based on the number of "yes" votes received. For example:

$$\begin{array}{cc} Votes & \begin{array}{ccc}\text{Decoy} & \text{Unknown} & \text{Desired}\end{array} \\ \begin{array}{c}0\\1\\2\\3\\4\end{array} & \begin{bmatrix}0.38 & 0.33 & 0.29\\ 0.35 & 0.34 & 0.31\\ 0.34 & 0.35 & 0.31\\ 0.33 & 0.34 & 0.33\\ 0.28 & 0.33 & 0.39\end{bmatrix} \end{array} \tag{15}$$

The probability of ID can then be computed at time k using Eq. 12 starting from an initial condition without bias such as: $P_{ID} = [0.33\ 0.33\ 0.33]$ for $c_i = [Decoy\ Unknown\ Desired]$.

The required voting is based on metrics specific to each type of attribute. In the case of $mass$, a distance metric is given by

$$\Delta mass_i = mass_{true} - mass_i \tag{16}$$

for track i and the desired target's "true" mass. If $|\Delta mass_i| < \varepsilon$, where ε is an empirical threshold, then track i receives a "yes" vote as a match.

Hue and saturation are represented by vectors of dimension n_{bins} x 1, lending themselves to a correlation metric. For each vector, the pairwise linear correlation coefficient is computed between the "true" and "track" hue and saturation vectors. A correlation coefficient arbitrarily close to one receives a "yes" vote as a match. The same approach is applied to the texture statistics, $[S(r),\ S(\theta)]$, requiring that the correlation between both measurements and their "true" values lie arbitrarily close to one in order to receive a "yes" vote for the texture component.

4.4 Value Function $f_{i,j}$

Obtaining the optimal solution relies upon using the available information to properly set the values that drive the integer program. The factors which influence the value function used in the integer program are based on the kinematics states of the tracking routine, the relative value of the segmentation methods, and the target ID associated with a given track. These are combined for each combination of target, sub-region, and segmentation using the expression:

$$f'_{i,j,k} = dist_{i,j,k} \cdot \kappa_k \cdot m_{j,k} \cdot \beta_j \tag{17}$$

Each of the coefficients will be described presently.

The value $dist_{i,j,k}$ represents the relative value of the target appearing at a given location. In order to generate a Gaussian-like weighting of the value, the error distance, $\rho^s_{i,j,k}$, is used as the argument of the error function given by:

$$dist_{i,j,k} = \frac{2}{\sqrt{\pi}} e^{(\rho^s_{i,j,k})^2} \int_{\rho^s_{i,j,k}}^{\infty} e^{-t^2} dt \tag{18}$$

Where

$$\rho^s_{i,j,k} = \sqrt{\left(p_{x_k} - s_{x_i}\right)^2 + \left(p_{y_k} - s_{y_i}\right)^2} \tag{19}$$

with s_{x_i} and s_{y_i} representing the center of sub-region i.

Next, the target ID is used to provide a scale factor, κ_k, is applied for target k in order to put emphasis on a particular type of classification. Based on a set of possible target ID's, the value κ_k may be defined as:

$$\kappa_k = \begin{cases} 2, & ID = 1 \\ 5, & ID = 2 \\ 3, & ID = 3 \end{cases} \tag{20}$$

The values for κ_k are arbitrary and represent an ad hoc tuning parameter within the value assignment constraint. A scaling $m_{j,k}$ is given by the ratio between the number of times any segmentation j has been performed on a given track k to the number of times any other operation has been performed to this same track. This weight can be illustrated as:

$$m_{j,k} = \begin{cases} 1, & & j = 1 \\ \frac{N_{2,k}}{N_{3,k}}, & N_{2,k} > N_{3,k}, & j = 2,3 \\ \frac{N_{3,k}}{N_{2,k}}, & N_{2,k} < N_{3,k}, & j = 2,3 \end{cases} \tag{21}$$

where $N_{j,k}$ is the number of times the j^{th} segmentation has been performed on the k^{th} target.

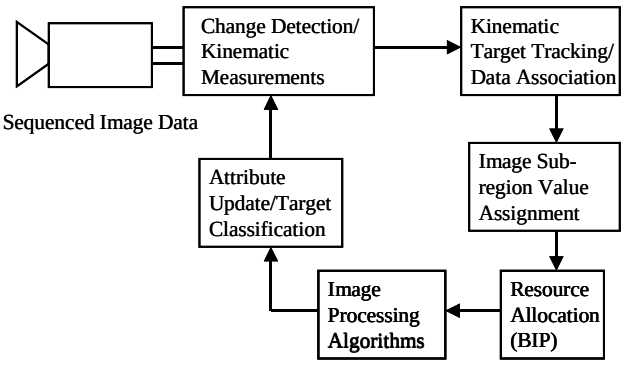

Figure 2: System with Binary Integer Program

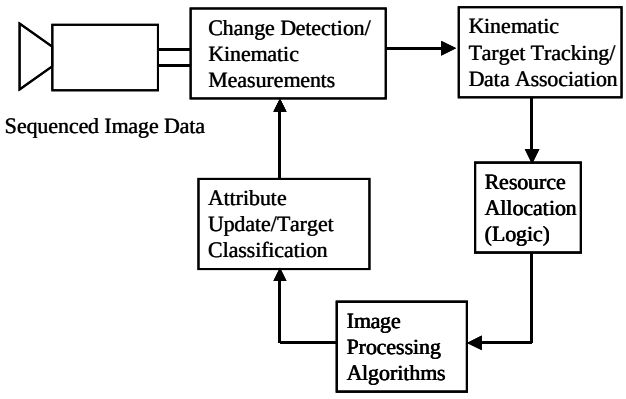

Figure 3: System with Logic Algorithm

For the three-segmentation case used here, a relative weight, β, is now applied to discern the benefit of performing one type of processing over the other in terms of the benefit obtained in classification that results from this type of processing. For example,

$$\beta_j = \begin{cases} 1, & j = 1, \quad mass \\ 2, & j = 2, \quad color \\ 3, & j = 3, \quad texture \end{cases}$$

The values β_j are arbitrary and represent an additional tuning parameter within the value assignment construct. Finally, since the goal is to determine a value for each sub-region and each segmentation algorithm, we will sum the values associated with each target using the expression:

$$f_{i,j} = \sum_k f'_{i,j,k} \tag{22}$$

to arrive at an objective function coefficient for each decision variable in the BIP.

5 System Overview

The goal of this paper is to achieve optimal resource allocation of the image processing resources available to the computer. The optimal solution itself incurs a computational cost, so an analysis of the performance of a system computing an optimal solution should be compared with another, simpler allocation method. Several components are used to combine the image processing, resource allocation, and target kinematic and attribute tracking. The program implemented divides the work into routines that perform each of the necessary tasks and provides information to the next task. This construct is illustrated in Figures 2 and 3. The purpose of the Logic Algorithm system is to provide a means to compare the optimal resource allocation system with a system that provides resource allocation by making simple logic decisions. The performance of the Integer Program system will be compared to the Logic system to determine its effectiveness in terms of its ability to provide proper target classification and reduced resource consumption.

6 Simulation Study

A simulation study was conducted containing three targets moving through a scene to illustrate the validity of the algorithms proposed in this paper. Two scenarios were run. One consisted of 57 time epochs (57 frames of imagery) and the other contained 48 time epochs. The targets are soldiers with different uniforms, moving across a fixed scene containing occlusions. Although all of the targets are very similar in size, their color and texture attributes differ somewhat. Two of the three targets are essentially distractors, and it is desirable to track the single high-valued target. The other targets (decoys) generate a computational load on the image processing, requiring some method for prioritizing the resource allocation. When targets enter an occluded region they no longer generate measurements or attribute data, but they may re-emerge at a later time. While it is desired to be able to establish a target's ID at a time before it enters an occlusion, the ultimate purpose of this study is to be able to establish a positive identification on the "high-valued" target when it emerges from an occlusion event. As the targets move through the scene, they are classified according to their attributes. There will be situations that may cause improper classification. Targets may move so close they merge onto each other, they may become partially occluded behind an obstacle, and their paths may cross. Simulations will be processed using two types of allocation algorithms that select which sub-regions to use for image processing. One model is driven by the binary integer program and will be referred to as "BIP". The other will make simple logic-based decisions driven solely by the targets' kinematic states. This will be referred to as "Logic," with the principal idea was to generate a competing algorithm for the BIP concept. The logic algorithm will order change detection around each target, and a combination of color and texture processing is accomplished in and around each change detection. The Logic concept represents a greedy algorithm, which is not constrained on a global level. In other words, it is allowed order processing (in the vicinity of a change) without regard for the total processing being conducted. In contrast, when multiple targets are competing for resources, the

Table 1: Resources Consumed by Allocation Methods

Scenario 1			
BIP	Mass	Color	Texture
Averages:	1809.8	187.9	337.3
Tokens:	1809.8	751.6	921
	Total	Tokens:	3482.4
Logic			
Averages:	1419.7	337.3	330.4
Tokens:	1419.7	1349.2	1652
	Total	Tokens:	4420.9
% Savings:	-27.4	44.2	44.2
	Net	Savings:	21.2%

Scenario 2			
BIP	Mass	Color	Texture
Averages:	2178.3	229.6	228
Tokens:	2178.3	918.4	1140
	Total	Tokens:	3482.4
Logic			
Averages:	1602.5	334.8	339.7
Tokens:	1602.5	1339.2	1698.5
	Total	Tokens:	4640.2
% Savings:	-35.9	21.4	32.9
	Net	Savings:	8.7%

BIP divides limited resources among the various targets on a global scale.

A short Monte Carlo analysis of 10 runs was performed to determine general characteristics of the two resource allocation methods. When using basic filtering on the images, it was determined that it takes about 5 times longer for a texture measurement and about 4 times longer for a color measurement than a mass measurement. A token is defined as the amount of time to perform a mass measurement. Therefore, for each segmentation, j, these relative token costs are called τ_j. For each segmentation that is performed v_j times per epoch, the total number of tokens, Ψ, required to perform an epoch's image processing is defined as:

$$\Psi = \sum_{j=1}^{3} \tau_j v_j \tag{23}$$

This will be used to determine the relative cost for the simulations. Table 1 shows the amount of resources consumed using each of the allocation methods for the two scenarios. From this table, it can be seen that the BIP algoritm uses fewer resources than the Logic algorithm. It must be recognized that the computational load of implementing the BIP algorithm is *not* included here.

The classification accuracy was computed for each scenario by counting how many times the correct or incorrect classification was applied to each visible target at each epoch. These totals are provided in Table 2. In scenario 1, there are 30 opportunities for a correct desired identification in each run, and 48 opportunities for a correct decoy identification. Scenario 2 contains 27 opportunities for a correct desired identification and 45 opportunities for a correct decoy identification.

Table 2: Classification Quantities

Scenario 1				
	True		False	
	Pos.	Neg.	Pos.	Neg.
BIP	73	426	54	227
Logic	43	435	45	257
Scenario 2				
BIP	100	391	59	170
Logic	63	424	26	207
	$C_D \vert T_D$	$C_N \vert T_N$	$C_D \vert T_N$	$C_N \vert T_D$

Table 3: Overall Classification Accuracy

	Scenario 1		
	BIP	Logic	Best
$P(T_D\|C_D)$	**0.575**	0.500	BIP
$P(T_N\|C_D)$	**0.425**	0.500	BIP
$P(T_D\|C_N)$	**0.348**	0.37	BIP
$P(T_N\|C_N)$	**0.652**	0.63	BIP
	Scenario 2		
$P(T_D\|C_D)$	0.629	**0.708**	Logic
$P(T_N\|C_D)$	0.371	**0.292**	Logic
$P(T_D\|C_N)$	**0.303**	0.328	BIP
$P(T_N\|C_N)$	**0.697**	0.672	BIP

Given the information presented in Table 2, it is possible to determine the probability of a target being of one of the two types, desired or decoy, given a classification. There are four permutations that can be measured. These are the probabilities of having desired or decoy targets, given that there is a positive or negative classification. Conditional probabilities can be calculated via:

$$P(T_x|C_y) = \frac{P(C_y|T_x)P(T_x)}{P(C_y)} \tag{24}$$

where T_x indicates a target truly being desired or decoy and C_y indicates a desired or decoy classification.

The values in Table 3 indicate the probabilities given by Eq. (24). The variable T_D indicates a (D)esired target, while T_N indicates the (N)on-desired or decoy target. The C_D indicates a (D)esired classification, while C_N indicates the (N)on-desired or decoy classification. The algorithm that provides the better performance for each case is indicated in the right column of each of the scenarios' information. The probabilities, $P(T_N)$ and $P(T_D)$ are determined by the total number of times each of the target types appears in the scene, divided by the total appearances of all targets. The probabilities $P(C_D)$ and $P(C_N)$ are the relative chance of each of the two classifications occurring, and are computed as finding the number of times each classification is made divided by the total number of classifications made.

7 Conclusions

Both resource allocation methods studied here provide acceptable and relatively similar results in terms of correctly classifying targets. The BIP method provides improved performance in terms of computational load over the simple Logic method. This benefit does come at a cost. A valuation function must accurately describe the value of collecting information on a given target, and a BIP that is too complex may not be able to determine resource allocation quickly. It is apparent that the study of BIP's choices can be used to drive the development of other logic-based algorithms that will function quickly as well as provide comparable results. Certainly, such an ad-hoc logic set can be employed, with adequate performance as demonstrated here. This leads to the concept of using the BIP to help in the development of logic-based algorithms for online use, which is the basis for future work in this area. A different method of approaching the problem may provide a more robust and reliable system. Such a resource allocation system would be two-layered. The first layer of the system would determine how many system resources at a given time can be allocated to various areas of the problem. This would be based on standard linear programming techniques and provide information such as what image processing algorithms to utilize and which targets should be examined. The second layer would use simple logic to determine where to apply the resources that are allowed at each epoch.

References

[1] Blackman, Samuel S. *Multiple-Target Tracking with Radar Applications.* Norwood, MA: Artech House, 1986.

[2] Winston, Wayne L. *Operations Research Applications and Algorithms.* Boston, MA: PWS-Kent, 1987.

[3] Gonzalez, Rafael C., et al. *Digital Image Processing Using MATLAB.* Upper Saddle River, NJ: Pearson Prentice Hall, 2004.

[4] Li, X. Rong and Vesselin P. Jilkov. "A Survey of Maneuvering Target Tracking: Dynamic Models," *Proceedings of SPIE Conference on Signal and Data Processing of Small Targets*, (April 2000).

[5] Blackman, Samuel S. and Robert Popoli. *Design and Analysis of Modern Tracking Systems.* Norwood, MA: Artech House, 1999.

[6] Bar-Shalom, Yaakov and Xiao-Rong Li. *Multitarget-Multisensor Tracking Principles and Techniques.* Stoors, CT: YBS Publishing, 1995.

[7] Leon-Garcia, Alberto. *Probability and Random Processes for Electrical Engineering.* Reading, MA: Addison-Wesley Publishing Company, 1994.